EMBEDDING HARD PHYSICAL CONSTRAINTS IN CONVOLUTIONAL NEURAL NETWORKS FOR 3D TURBULENCE

Arvind T. Mohan *
Center for Nonlinear Studies
Computer, Computational and Statistical Sciences Division
Los Alamos National Laboratory
Los Alamos, NM, United States
arvindm@lanl.gov

Nicholas Lubbers & Daniel Livescu
Computer, Computational and Statistical Sciences Division
Los Alamos National Laboratory
Los Alamos, NM, United States
{nlubbers,livescu}@lanl.gov

Michael Chertkov
Dept. of Mathematics
University of Arizona
Tucson, AZ, United States
chertkov@math.arizona.edu

ABSTRACT

Deep learning approaches have shown much promise for physical sciences, especially in dimensionality reduction and compression of large datasets. A major issue in deep learning of large-scale phenomena, like fluid turbulence, is the lack of physical guarantees. In this work, we propose a general framework to directly embed the notion of incompressible fluids into Convolutional Neural Networks, for coarse-graining of turbulence. These **physics-embedded neural networks** leverage interpretable strategies from numerical methods and computational fluid dynamics to enforce physical laws and boundary conditions by taking advantage the mathematical properties of the underlying equations. We demonstrate results on 3D fully-developed turbulence, showing that the *physics-aware inductive bias* drastically improves local conservation of mass, without sacrificing performance according to several other metrics characterizing the fluid flow.

1 INTRODUCTION

A revolution is underway in physical and computational sciences with the promise of neural network (NNs) approaches in modeling unresolved physics, accurate data compression and model reduction. An important component is fluid mechanics, where the curse of dimensionality has hindered much progress. There are two key issues with learning high dimensional data: 1) The computational/memory limitations in employing enough training parameters 2) The black-box nature of NNs that do not guarantee physical conservation laws and boundary conditions (BCs). An important example is the continuity equation, which for incompressible fluids, becomes the divergence-free condition for the velocity field V:

$$\nabla \cdot V = 0 \tag{1}$$

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies. Funding acknowledgements go at the end of the paper.

Recent approaches (Raissi et al., 2019; Wu et al., 2019) to incorporate it relies on penalizing the network in the loss function to encourage solutions to obey Eqn. 1 as well as known BCs. We call this approach a *soft constraint*, and the weight of the soft constraint regularization becomes an additional training hyperparameter. While soft constraints have been popular, they provide no guarantees due to lack of *inductive bias*(Gaier & Ha (2019); Wang et al. (2019)) in the model. In this paper, we report first attempts on a general methodology to address these challenges in a Convolutional Neural Network (CNN) framework with strong inductive bias. We apply this to a high-fidelity Direct Numerical Simulation (DNS) of a 3D Homogeneous Isotropic turbulence (HIT) flow, described in Appendix A. Our approach also adds explainability by interpreting time-tested strategies from numerical methods and CFD as specific instances of CNN kernels, without any additional trainable parameters.

2 Embedding Physical Operators in Convolutional Neural Networks

We adopt the philosophy of most PDE solvers, where conservation laws and BC constraints are *strictly enforced at all times*, rather than penalizing them separately. A core aspect of this is an accurate and unambiguous definition of the operator $\nabla\cdot$ which is also amenable to the backpropagation. Backpropagation through the physics operators and BC creates a strong inductive bias for the NN. There are two major challenges: **First**, Constructing spatial derivatives for differential operators (i.e. $\nabla\times$, $\nabla\cdot$, $\nabla^2\cdot$, etc.) that are compatible with the backpropagation. **Second**, Enforcing BCs for the velocity fields. We now present our approach to address these challenges.

2.1 Spatial Derivative Computation in CNN Kernels

CNNs are used to learn the spatial features with a convolution kernel f on a domain of interest g at a layer n. The n^{th} CNN layer computes $y_n = f * g_{n-1}$, where g_{n-1} is the output of the $(n-1)^{th}$ layer. Therefore, at layer $n+1$, $y_{n+1} = f * g_n$. The kernel translation is also an important hyperparameter, called *striding*, that can be performed for every point in the mesh (1-step), or by skipping over a two points (2-step). To compute derivatives of field ϕ on a discretized mesh, we adopt strategies from well-known finite difference (FD)/Finite volume (FV) numerical methods, which are analytically derived from Taylor series expansions (Ferziger, 1981; Spalding, 1972). For a standard 2$^{\text{nd}}$ order central difference FV scheme shown in Eqn. 2 (left), Its coefficients can be expressed in matrix form, called a *numerical stencil* (right).

$$\frac{\partial \phi}{\partial x} = \frac{\phi(x+\delta x)-\phi(x-\delta x)}{2\delta x} + O(\delta x)^2 \qquad \longleftrightarrow \qquad \frac{\partial \phi}{\partial x} = \begin{bmatrix} -\frac{1}{2\delta x} & 0 & \frac{1}{2\delta x} \\ -\frac{1}{2\delta x} & 0 & \frac{1}{2\delta x} \\ -\frac{1}{2\delta x} & 0 & \frac{1}{2\delta x} \end{bmatrix} \tag{2}$$

CNN kernels are known to be *structurally equivalent* to numerical stencils (Long et al., 2017; Dong et al., 2017). FV stencils are CNN kernels with fixed, non-trainable weights that compute a derivative to the desired order of accuracy, since both are *mathematically identical* for 1-step striding. If f is the FV kernel, and g the numerical mesh, the derivative is $\frac{\partial \phi}{\partial x} = f * g$, at layer n. This simple, but powerful, connection allows us to embed these stencils as CNN layers to compute our derivatives of interest, while simultaneously being *interpretable*.

2.2 Enforcing Periodic Boundary Conditions

The HIT flow has spatially periodic BCs in all three directions and we present here a method to rigorously enforce these in CNNs, to a desired order of discretization accuracy. Figure 1 shows the aforementioned CNN stencil kernel on a mesh. The $(3\times3\times3)$ kernel performs convolution on ϕ and the outermost column/row of cells in the mesh are forfeited. A popular CNN fix is to "zero pad" the boundaries, but this does not enforce BCs and leads to inaccuracies in subsequent derivatives. We resolve this discrepancy while simultaneously satisfying the BCs, by employing *Ghost cells* (Fadlun et al., 2000; Tseng & Ferziger, 2003) from CFD. Ghost cells are "virtual" cells which are defined at

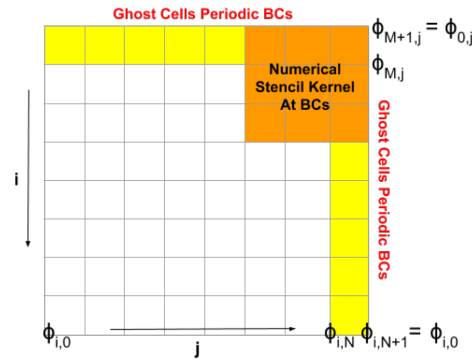


Figure 1: Periodic BCs enforced as ghost cell padding in CNNs (k^{th} direction in 3D not shown)

mesh boundaries, so that a derivative of the desired order consistent with the numerical stencil can be computed. Periodic BCs imply the flow leaving the domain in one direction enter the domain in the opposite direction. In Fig. 1, we pad with Ghost cells ($N+1, M+1, L+1$) to mimic this behavior, and the solutions $\phi_{i,N+1,k} = \phi_{i,0,k}$, $\phi_{M+1,j,k} = \phi_{0,j,k}$ and $\phi_{i,j,L+1} = \phi_{i,j,0}$ then exactly satisfy the periodic BCs in the CNN.

3 Physics Embedded CNN Architecture with Hard Constraints

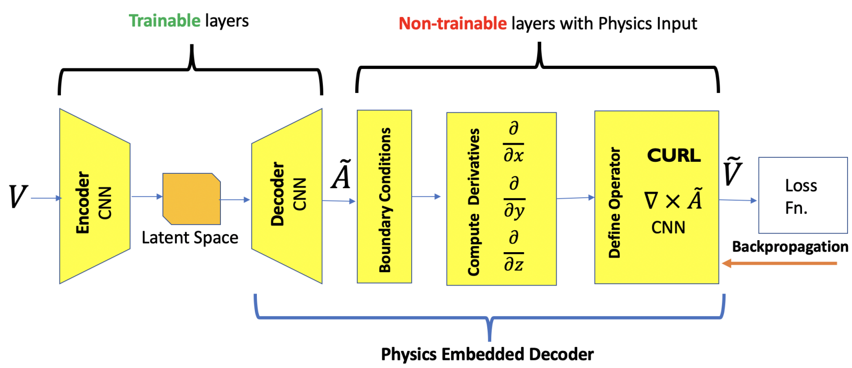


Figure 2: Physics Embedded Convolutional Autoencoder (PhyCAE) with hard divergence-free constraints for coarse grained $\tilde{V}$

For incompressible turbulence, a potential formulation (Hirasaki & Hellums, 1968; Morino, 1985) based on the Helmholtz decomposition with vector potential A and scalar potential ψ governs the flow. For the data analyzed here, the boundary conditions are periodic, so $\psi = 0$ is a valid solution.

$$V = \nabla \times A + \cancelto{0}{\nabla\psi} \tag{3}$$

The key idea is as follows: Instead of only predicting a velocity field, we choose to make an intermediate prediction for coarse-grained vector potential $\tilde{A}$, while framing the final prediction $\tilde{V}$ in the target velocity space via Eqn. 3, implemented as a numerical stencil. Then, predictions $\tilde{V}$ will *automatically* obey Eqn. 1 up to the accuracy of the stencil since $\nabla \cdot V = \nabla \cdot (\nabla \times A) = 0$. Figure 2 shows the autoencoder with a **physics-embedded CNN AutoEncoder** (PhyCAE) where this strategy is implemented. We can constrain the network to implicitly learn $\tilde{A}$ by requiring that $\nabla \times$ of the decoder prediction $\tilde{A}$ be equal to $\tilde{V}$. The ghost cell padding layer enforces BCs and the next layer computes $\nabla \times$ on the $\tilde{A}$ field. Therefore, all layers after the decoder CNN in the PhyCAE are **non-trainable, transparent and interpretable**, as they are constructed with numerical methods.

4 RESULTS

We train two cases: a) Standard CAE with zero padding, and b) PhyCAE which comprises of the standard CAE with the same hyperparameters (Appendix A.1), but with the addition of the physics embedded layers in Fig. 2. One of the key expectations from any hard constraint is that the network must be cognizant of the imposed physics and BCs *from the very first epoch*. To quantify how well the constraint is realized, we measure the total absolute divergence (TAD) across each sample averaged over the samples, given by $\sum |\nabla \cdot \tilde{V}|$. Note that TAD is not exactly zero due to discretization and single-precision arithmetic. Figure 3 shows the network TAD on the training data as a function of training epochs for both CAE and PhyCAE. For CAE, we see a spike in TAD as high as $\sim 10^{-1}$, and approaches $\sim 10^{-2}$, while the PhyCAE starts at $\sim 10^{-2}$ and trends downward, even oscillating near numerical zero, and settles between 10^{-4} and 10^{-5}. In other words, the *best-case* for the CAE is comparable to the *worst-case* for the PhyCAE. Even for test data, the PhyCAE TAD is more than 2 orders of magnitude better than CAE, further emphasizing robustness of the physics embeddings. We now compare 2 important tests of turbulence as diagnostic metrics for the accuracy of the coarse-grained flow(Appendix B). First, the *Kolmogorov energy spectra* in Figure 4a shows the spectra of the PhyCAE and CAE $\tilde{V}$ compared with the DNS test data V. The results show excellent large scale (low wavenumbers) and inertial range accuracy by the PhyCAE, very similar to that of CAE. Second, given by *probability density functions of velocity gradients* is studied in Fig. 4b. We see

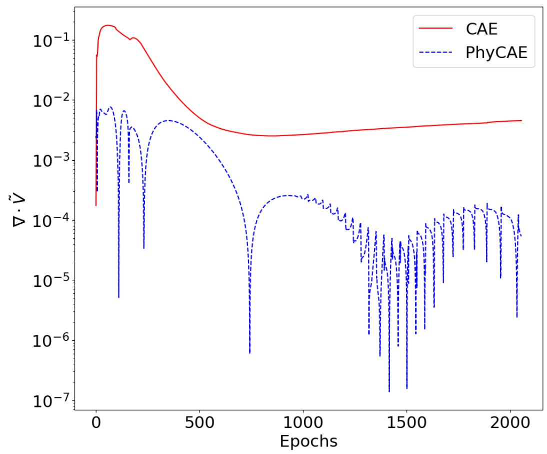
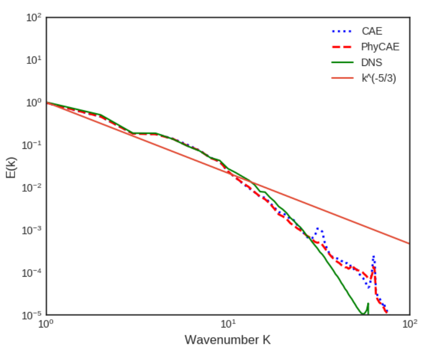
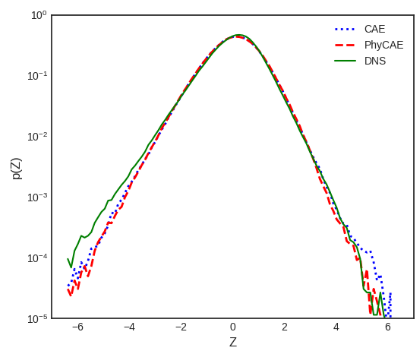


Figure 3: Variation of $\nabla \cdot \tilde{V}$ with training epochs for CAE vs. PhyCAE. **Final Divergence:** $\approx \mathbf{10^{-5}}$ for PhyCAE vs $\approx \mathbf{10^{-2}}$ for CAE



(a) Kolmogorov Energy Spectra: CAE and PhyCAE vs DNS



(b) PDFs of velocity gradient: CAE and PhyCAE vs DNS

Figure 4: Results: PhyCAE $\tilde{V}$ vs. CAE $\tilde{V}$

excellent matches between PhyCAE $\tilde{V}$ and DNS, with PhyCAE being slightly more accurate than CAE in the tails. Most of the discrepancies are localized at the small scales (high wavenumbers), due to the information loss that occurs during coarse graining. This is an acceptable trade-off since most practical applications of ROMs focus only on large/inertial scales, which are modeled well.

5 CONCLUSION

Data from physical phenomena is extremely high dimensional and necessitates compression and model reduction for timely, efficient analysis and insights (Overpeck et al., 2011). Another major application is surrogate modeling and super-resolution of high-fidelity turbulence (San & Maulik, 2018). Incompressible turbulence is commonplace in several phenomena, and a common difficulty in ML for these flows is ensuring mass conservation is strictly obeyed, i.e. $\nabla \cdot V = 0$, without increasing computational costs. This work introduces a structural and interpretable method of enforcing such laws in CNN architecture as a hard constraint, without additional hyperparameters to tune. The approach can be extended to general constraints on CNNs of form $L(V) = 0$ for differential operators L and fields V, by defining FV stencils of the appropriate order for L. The physics-aware inductive bias of this CNN allows it to perform far better than vanilla CNN while training with the identical hyperparameters, *without any increase* in the number of trainable parameters.

ACKNOWLEDGMENTS

This work has been authored by employees of Triad National Security, LLC which operates Los Alamos National Laboratory (LANL) under Contract No. 89233218CNA000001 with the U.S. Department of Energy/National Nuclear Security Administration. Authors have been supported by LANL's LDRD program, project number 20190058DR. Author 1 also thanks the Center for Nonlinear Studies at LANL for support and acknowledges the ASC/LANL Darwin cluster for GPU computing infrastructure.

REFERENCES

Don Daniel, Daniel Livescu, and Jaiyoung Ryu. Reaction analogy based forcing for incompressible scalar turbulence. *Physical Review Fluids*, 3(9):094602, 2018.

Bin Dong, Qingtang Jiang, and Zuowei Shen. Image restoration: Wavelet frame shrinkage, nonlinear evolution pdes, and beyond. *Multiscale Modeling & Simulation*, 15(1):606–660, 2017.

EA Fadlun, R Verzicco, Paolo Orlandi, and J Mohd-Yusof. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of computational physics*, 161(1):35–60, 2000.

Joel H Ferziger. *Numerical methods for engineering application*, volume 1. Wiley New York, 1981.

Adam Gaier and David Ha. Weight agnostic neural networks. *arXiv preprint arXiv:1906.04358*, 2019.

Oliver Hennigh. Lat-net: compressing lattice boltzmann flow simulations using deep neural networks. *arXiv preprint arXiv:1705.09036*, 2017.

GEORGE JIRO Hirasaki and JD Hellums. A general formulation of the boundary conditions on the vector potential in three-dimensional hydrodynamics. *Quarterly of Applied Mathematics*, 26(3): 331–342, 1968.

D Livescu, J Mohd-Yusof, MR Petersen, and JW Grove. Cfdns: a computer code for direct numerical simulation of turbulent flows. *Los Alamos National Laboratory Technical Report No. LA-CC-09-100*, 2009.

Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. Pde-net: Learning pdes from data. *arXiv preprint arXiv:1710.09668*, 2017.

Lu Lu, Xuhui Meng, Zhiping Mao, and George E Karniadakis. Deepxde: A deep learning library for solving differential equations. *arXiv preprint arXiv:1907.04502*, 2019.

Arvind Mohan, Don Daniel, Michael Chertkov, and Daniel Livescu. Compressed convolutional lstm: An efficient deep learning framework to model high fidelity 3d turbulence. *arXiv preprint arXiv:1903.00033*, 2019.

Luigi Morino. Scalar/vector potential formulation for compressible viscous unsteady flows. *NASA Technical Reports*, 1985.

Jonathan T Overpeck, Gerald A Meehl, Sandrine Bony, and David R Easterling. Climate data challenges in the 21st century. *science*, 331(6018):700–702, 2011.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Omer San and Romit Maulik. Extreme learning machine for reduced order modeling of turbulent geophysical flows. *Physical Review E*, 97(4):042322, 2018.

Dudley Brian Spalding. A novel finite difference formulation for differential expressions involving both first and second derivatives. *International Journal for Numerical Methods in Engineering*, 4 (4):551–559, 1972.

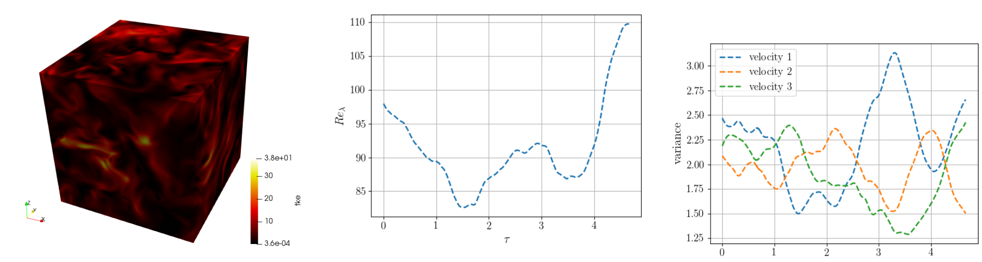
Yu-Heng Tseng and Joel H Ferziger. A ghost-cell immersed boundary method for flow in complex geometry. *Journal of computational physics*, 192(2):593–623, 2003.

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. *arXiv preprint arXiv:1911.08655*, 2019.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

Jin-Long Wu, Karthik Kashinath, Adrian Albert, Dragos Chirila, Heng Xiao, et al. Enforcing statistical constraints in generative adversarial networks for modeling chaotic dynamical systems. *arXiv preprint arXiv:1905.06841*, 2019.

A 3D Homogeneous Isotropic Turbulence Dataset and Training



(a) Instantaneous turbulent kinetic energy (b) Reynolds number (based on Taylor microscale) (c) Individual velocity variances

Figure 5: Representative Statistics of the Simulation

The dataset consists of a 3D Direct Numerical Simulation (DNS) of homogeneous, isotropic turbulence, in a box of size 128^3. We denote this dataset as *HIT* for the remainder of this work. We provide a brief overview of the simulation and its physics in this section, and a detailed discussion can be found in Daniel et al. (2018). The ScalarHIT dataset is obtained using the incompressible version of the CFDNS Livescu et al. (2009) code, which uses a classical pseudo-spectral algorithm. We solve the incompressible Navier-Stokes equations:

$$\partial_{x_i} v_i = 0, \qquad \partial_t v_i + v_j \partial_{x_j} v_i = -\frac{1}{\rho}\partial_{x_i} p + \nu \Delta v_i + f_i^v,$$

where f^v is a low band forcing, restricted to small wavenumbers $k < 1.5$ [1]. The 128^3 pseudo-spectral simulations are dealiased using a combination of phase-shifting and truncation to achieve a maximum resolved wavenumber of $k_{max} = \sqrt{2}/3 \times 128 \sim 60$.

For illustration, Figure 5a shows the turbulent kinetic energy at a time instant. Figure 5b shows the variation in the Taylor-microscale based Reynolds number with the eddy turnover time, which characterizes the large turbulence scales. Finally, the variances in all 3 velocity components are shown in Fig. 5c. Based on the sampling rate, each eddy turnover time τ consists of 33 snapshots. The training dataset uses 22 snapshots $\approx 0 - 0.75\tau$ and test dataset also consists of 22 snapshots in $\approx 4 - 4.75\tau$.

A.1 Training Details and Extensions

The CAE architecture has 3 layer encoder-decoder with an ADAM optimizer and L2 loss, with only 6 filters at each level to avoid over-fitting and study the effects of inductive bias. In CAE, the compression ratio between the dimension of a single datapoint (3×128^3) and that of the latent space (6×15^3), is ≈ 300. We remark that much like any PDE solver, the discretization errors affect the accuracy of the hard constraint. Due to the interpretable hard-constraint approach of the PhyCAE, this could be further decreased by improving the spatial discretization method in Eqn. 2 from a 2nd to a higher order scheme. This extension is straightforward since the CNN allows for kernels of larger sizes produced by higher order numerical schemes. This would require a corresponding change in number of ghost cells, which can be implemented as outlined in Section 2.2.

B Some Diagnostic Tests of Turbulence

We now briefly describe 2 basic tests of 3D turbulence which are used as "diagnostic" metrics in this work, for the accuracy of the flow predicted by the trained model.

B.1 4/5 Kolmogorov law and the Energy Spectra

The main statement of the Kolmogorov theory of turbulence is that asymptotically in the inertial range, i.e. at $L \gg r \gg \eta$, where L is the largest, so-called energy-containing scale of turbulence and η is the smallest scale of turbulence, so-called Kolmogorov (viscous) scale, $F(r)$ does not depend on r. Moreover, the so-called $4/5$-law states for the third-order moment of the longitudinal velocity increment

$$L \gg r \gg \eta : \quad S_3^{(i,j,k)} \frac{r^i r^j r^k}{r^3} = -\frac{4}{5}\varepsilon r, \tag{4}$$

where $\varepsilon = \nu D_2^{(i,j;i,j)}/2$ is the kinetic energy dissipation also equal to the energy flux.

Self-similarity hypothesis extended from the third moment to the second moment results in the expectation that within the inertial range, $L \gg r\eta$, the second moment of velocity increment scales as, $S_2(r) \sim v_L (r/L)^{2/3}$. This feature is typically tested by plotting the energy spectra of turbulence (expressed via $S_2(r)$) in the wave vector domain, e.g. as shown in the results section.

B.2 Intermittency of Velocity Gradient

Consequently from Eqn. 4, the estimation of the moments of the velocity gradient results in

$$D_n \sim \frac{S_n(\eta)}{\eta^n}. \tag{5}$$

This relation is strongly affected by intermittency for large values of n (i.e. extreme non-Gaussian behavior) of turbulence, and is a valuable test of small scale behavior.